Scalable Nonparametric Sampling from Multimodal Posteriors with the Posterior Bootstrap

Edwin Fong, Simon Lyddon, Chris Holmes

Michael Komodromos

June 9, 2021

Dirichlet Distribution, definition

Dirichlet distribution: multivariate generalisation of the Beta distribution.

pdf is given by

$$f(x; \alpha) = \frac{1}{B(\alpha)} \prod_{i=1}^{p} x_i^{\alpha_i - 1} \quad (1)$$

where $x \in \mathbb{R}^p, \sum_i x_i = 1$ and $\alpha \in \mathbb{R}^p, a_i > 0$, and $B(\alpha)$ is a normalisation term.

Dirichlet Distribution, visualisation

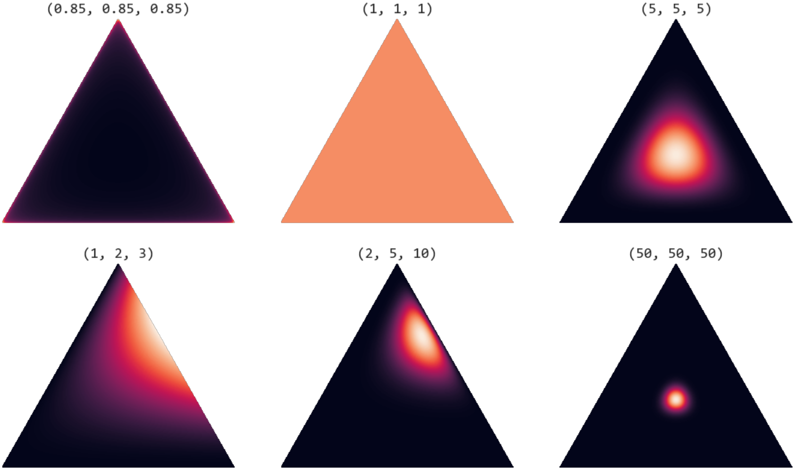


Figure: source:
https://towardsdatascience.com/dirichlet-distribution-a82ab942a879

Dirichlet Processes, definition

Dirichlet Process: a stochastic process where a finite subset of random variables have a Dirichlet distribution.

Dirichlet processes are specified by a base probability distribution, H and a concentration parameter α. Then for some finite disjoint partition of $S = \{B\}_{i=1}^{n}$ we have

$$(X_{B_1}, \ldots, X_{B_n}) \sim \text{Dirichlet}(\alpha H(B_1), \ldots, \alpha H(B_n)) \tag{2}$$

Dirichlet Processes, some intuition

What is a die?

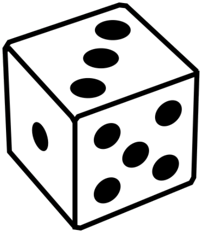
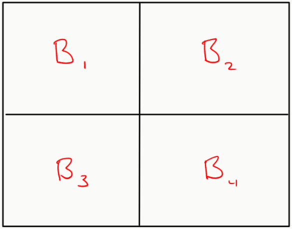


Figure: A Statistician's best friend (sorry coins).

Dirichlet Processes, some intuition

Sampling form a DP has the bi-product of a new probability distribution.

1. Partition a space



2. Compute some parameters

$$\mathrm{Dir}(\alpha H(B_1), \ldots, \alpha H(B_4))$$

3. Sample some weights

$$(X_{B_1}, X_{B_2}, X_{B_3}, X_{B_4})$$

Questions / Comments?

And then we're onto the paper!

Bayesian Non-parametric learning

Let $y = (y_1, \ldots, y_n)$ where $y_i \stackrel{iid}{\sim} F_0$ and let $\theta \in \Theta \subseteq \mathbb{R}^p$ be a parameter that indexes a family of probability distribution $\mathcal{F}_\Theta$.

Bayesian Non-parametric learning

Let $y = (y_1, \ldots, y_n)$ where $y_i \overset{iid}{\sim} F_0$ and let $\theta \in \Theta \subseteq \mathbb{R}^p$ be a parameter that indexes a family of probability distribution $\mathcal{F}_\Theta$.

Then we are interested in

$$\theta_0(F_0) = \arg\min_\theta \int l(y;\theta) dF_0(y) \tag{3}$$

where $l(y;\theta)$ is a loss function. For example if $l = (y-\theta)^2$ we would recover the mean.

Bayesian Non-parametric learning, cont.

The issue is we don't know F_0. So, what do Bayesian's do when they don't know something?

Bayesian Non-parametric learning, cont.

The issue is we don't know F_0. So, what do Bayesian's do when they don't know something?

Put a prior on it

Bayesian Non-parametric learning, cont.

The issue is we don't know F_0. So, what do Bayesian's do when they don't know something?

Put a prior on it

Putting a DP prior on F_0

$$F|a, F_\pi \sim DP(a, F_\pi) \tag{4}$$

Bayesian Non-parametric learning, cont.

And via the nice conjugacy properties of the DP the posterior $F|y_{1:n}$ is given as

Bayesian Non-parametric learning, cont.

And via the nice conjugacy properties of the DP the posterior $F|y_{1:n}$ is given as

$$F|y_{1:n} \sim DP(a+n, G_n) \tag{5}$$

Bayesian Non-parametric learning, cont.

And via the nice conjugacy properties of the DP the posterior $F|y_{1:n}$ is given as

$$F|y_{1:n} \sim DP(a+n, G_n) \tag{5}$$

where

$$G_n = \frac{a}{a+n}F_\pi + \frac{1}{a+n}\sum_i \delta_{y_i} \tag{6}$$

Bayesian Non-parametric learning, cont.

Finally our NPL posterior $\pi(\theta|y_{1:n})$ is given by

Bayesian Non-parametric learning, cont.

Finally our NPL posterior $\pi(\theta|y_{1:n})$ is given by

$$\pi(\theta|y_{1:n}) = \int \pi(\theta|F) d\pi(F|y_{1:n}) \tag{7}$$

Bayesian Non-parametric learning, cont.

Finally our NPL posterior $\pi(\theta|y_{1:n})$ is given by

$$\pi(\theta|y_{1:n}) = \int \pi(\theta|F)d\pi(F|y_{1:n}) \tag{7}$$

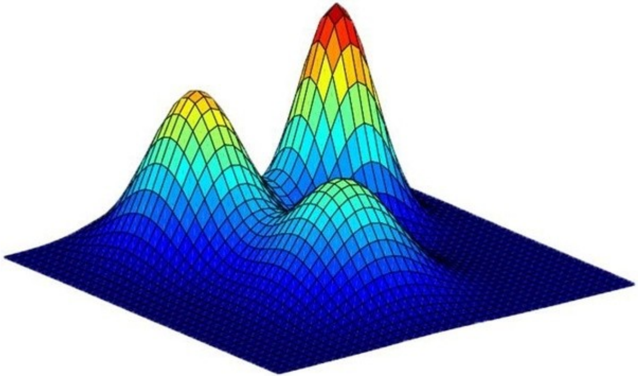
We can sample from $\pi(\theta|y_{1:n})$ using the following algorithm

Algorithm 1 NPL Posterior Sampling

for $i = 1$ **to** B **do**
  Draw $F^{(i)} \sim \mathrm{DP}(\alpha + n, G_n)$
  $\theta^{(i)} = \arg\min_\theta \int l(y, \theta)dF^{(i)}(y)$
end for

Gaussian Mixture Models

Weighted mixture of Gaussians



Gaussian Mixture Models, cont.

To use Bayesian Non-parametric learning all we need to do is define a loss function

$$l(y, \pi, \mu, \sigma) = -\log \sum_{=1}^{K} \pi_k \mathcal{N}(y; \mu_k, \text{diag}(\sigma_k^2)) \tag{8}$$

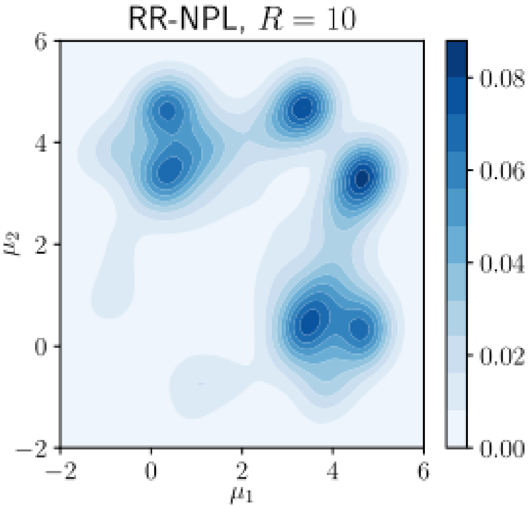
GMMs have multi-modal posteriors.

Gaussian Mixture Models, cont.

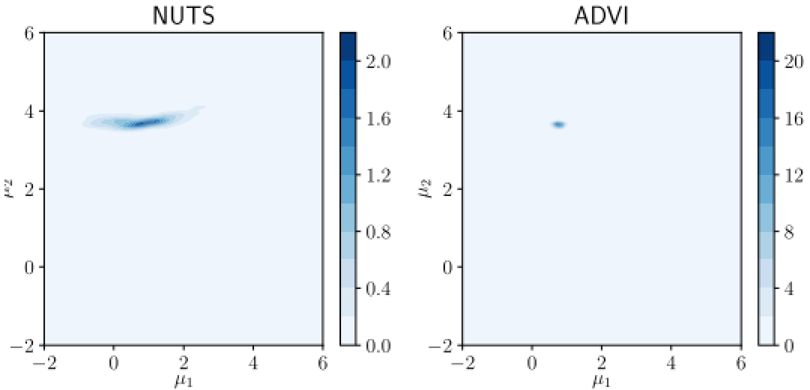
For the example, there are $K = 3$ groups, with

$$\begin{aligned} \pi &= (0.1, 0.3, 0.6) \\ \mu_0 &= (0, 2, 4) \\ \sigma_0^2 &= (1, 1, 1) \end{aligned}$$

Gaussian Mixture Models, cont.



Gaussian Mixture Models, cont.



Gaussian Mixture Models, discussion

NPL recovers the multi-modality of the posterior. But is this answer useful?

Gaussian Mixture Models, discussion

NPL recovers the multi-modality of the posterior. But is this answer useful?

Yes! We now know the problem is multi-modal... and can quantify the uncertainty about those modes

Gaussian Mixture Models, discussion

NPL recovers the multi-modality of the posterior. But is this answer useful?

Yes! We now know the problem is multi-modal... and can quantify the uncertainty about those modes

No! We might not care about those modes, but getting something useful out of the posterior. It's not really clear how we can do that.

Aren't DPs infinite dimensional?

DPs are infinite dimensional objects, computers are not.

Aren't DPs infinite dimensional?

DPs are infinite dimensional objects, computers are not.

In practice we need to truncate somewhere

Aren't DPs infinite dimensional?

DPs are infinite dimensional objects, computers are not.

In practice we need to truncate somewhere But that means we're going to be approximating

$$\pi(\theta|y_{1:n}) = \int \pi(\theta|F) d\pi(F|y_{1:n}) \tag{9}$$

Approximate posterior sampling

Algorithm 2 Posterior Bootstrap Sampling

Define T as truncation limit

Observed samples are $y_{1:n}$

for $i = 1$ **to** B **do**

Draw prior pseudo-samples $\tilde{y}^{(i)}_{1:T} \overset{iid}{\sim} F_\pi$

Draw $(w^{(i)}_{1:n}, \tilde{w}^{(i)}_{1:T}) \sim \mathrm{Dir}\,(1, \ldots, 1, \alpha/T, \ldots, \alpha/T)$

$$\theta^{(i)} = \arg\min_\theta \left\{ \sum_{j=1}^n w^{(i)}_j l(y_j, \theta) + \sum_{k=1}^T \tilde{w}^{(i)}_k l(\tilde{y}^{(i)}_k, \theta) \right\}$$

end for

What if I want to quantify uncertainty about a mode?

Randomness is introduced through the weights and psuedo-samples. Fixing our starting point θ_i allows us to explore the area around a mode

What if I want to quantify uncertainty about a mode?

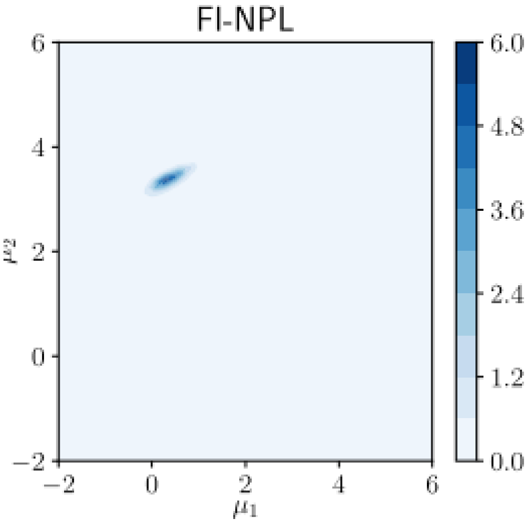
Randomness is introduced through the weights and psuedo-samples. Fixing our starting point θ_i allows us to explore the area around a mode

Algorithm 4 FI-NPL Posterior Sampling

Select θ^{init} from mode of interest
for $i = 1$ **to** B **do**
  Draw $F^{(i)} \sim \text{DP}(\alpha + n, G_n)$
  $\theta^{(i)} = \text{local arg min}_\theta \left(\int l(y, \theta) dF^{(i)}(y), \theta^{\text{init}} \right)$
end for

Back to GMMs

Back to GMMs



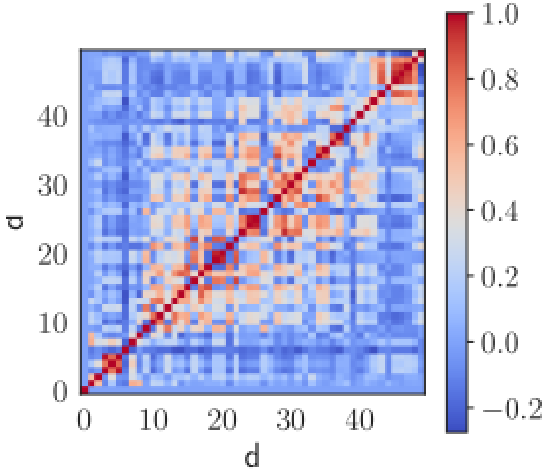
Example #2

Analysis of genotype / psuedo phenotype dataset, where the phenotype was generate by

$$y_i \sim \mathrm{Bernoulli}(\sigma(\beta^\top x_i)) \tag{10}$$

where $\beta \in \mathbb{R}^{50}$ with 5 randomly selected non-zero components.

Correlation Matrix



Inducing sparsity through the loss function

Inducing sparsity through the loss function

We can quantify uncertainty about parameters, but all this happens through our loss function.

Inducing sparsity through the loss function

We can quantify uncertainty about parameters, but all this happens through our loss function.

So all we need to do is have a sparsity inducing loss function.

$$l(y,\theta) - \log f_\theta(y) + \gamma g(\theta) \tag{11}$$

where f_θ is our likelihood and g is a penalisation term

Inducing sparsity through the loss function

We can quantify uncertainty about parameters, but all this happens through our loss function.

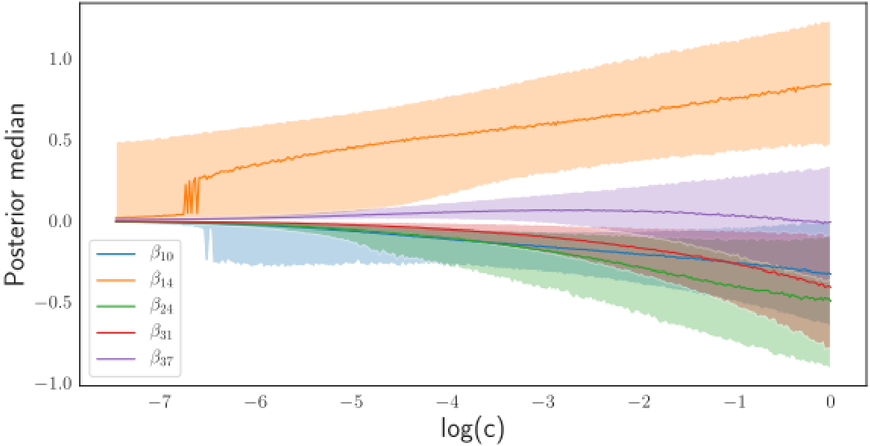
So all we need to do is have a sparsity inducing loss function.

$$l(y, \theta) - \log f_\theta(y) + \gamma g(\theta) \tag{11}$$

where f_θ is our likelihood and g is a penalisation term

Example: setting $g(\theta) = |\theta|$ gives the Bayesian NPL-Lasso

Example #2, results



Conclusions

A cool idea for uncertainty quantification.

Priors aren't really a thing, it's all done via the loss function

Captures multi-modality

Quick if we have a lot of compute, sampling from the NPL posterior can be done in parallel

Thanks for Listening

Questions?